ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

GENERACIÓN DE UNA INTERFAZ DE ENTRADA/SALIDA DE DATOS PARA UN SOFTWARE DE CÁLCULO ELECTROMAGNÉTICO

(Generation of an Input/Output Data Interface for an Electromagnetic Calculation Software)

Para acceder al Título de

Graduado en
Ingeniería de Tecnologías de Telecomunicación

Autor: Alejandro Gutiérrez Camacho

Febrero - 2019

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por:
Director del TFG:
Título: " "
Title: " "

Presentado a examen el día:

para acceder al Título de

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente (Apellidos, Nombre):
Secretario (Apellidos, Nombre):
Vocal (Apellidos, Nombre):

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente

Fdo.: El Secretario

Fdo.: El Vocal

Fdo.: El Director del TFG
(sólo si es distinto del Secretario)

Vº Bº del Subdirector

Trabajo Fin de Grado Nº (a asignar por Secretaría)

Agradecimientos

En primer lugar, a Luis Valle porque sin él no podría estar escribiendo este trabajo, gracias por toda la ayuda y siempre tener buenos consejos y sugerencias.

Después querría agradecérselo a mis pilares fundamentales, mis padres y mis hermanos ya que sin ellos no habría sido posible, como no hubiese sido posible tampoco sin el apoyo de mis amigos, los de siempre.

También me gustaría agradecérselo a Javier Guitian, debido a que desde que comenzó esta aventura no paró de decirme que era capaz de sacar cualquier cosa.

Por último, agradecer a todos los que habéis formado parte de esta etapa, estéis o no ahora mismo conmigo, gracias.

Resumen

En este trabajo se presenta una aplicación que hace uso de un programa que resuelve problemas electromagnéticos.

En primer lugar, se comentan las cosas más importantes sobre modelado geométrico, los tipos de modelado más importantes que existen y la elección del que se aplica en el programa, una vez visto esto pasamos a analizar que curvas se utilizan y que superficies se utilizan, dando las características básicas y porque utilizamos dichas superficies y curvas.

Después pasamos a analizar las técnicas numéricas que utiliza el programa en este caso la más importante y sobre lo que se basa el funcionamiento es el Método de los Momentos, se darán las características más importantes y se tocarán temas de las superficies que se utilizan para ver cómo con este método podemos dividirlas y que sea lo más óptimo para lo que se busca.

Una vez hemos explicado todo esto, pasamos a lo que de verdad es el objetivo del trabajo y es la aplicación, se explicará de que se compone la aplicación, con que ha sido creada, como funciona y como es la estructura de los datos.

Abstract

In this work an application is presented that makes use of a program that solves electromagnetic problems.

First, we discuss the most important things about geometric modeling, the most important types of modeling that exist and the choice of what is applied in the program, once we have seen this, we analyze which curves are used and which surfaces are used, giving the basic characteristics and because we use these surfaces and curves.

After we analyze the numerical techniques that the program uses in this case the most important one and what the operation is based on is the Method of Moments, the most important characteristics will be given and topics of the surfaces that are used to see will be touched how with this method we can divide them and that is the most optimal for what is sought.

Once we have explained all this, we turn to what is really the objective of the work and is the application, it will explain what the application is composed of, how it has been created, how it works and how is the structure of the data.

Índice

Capítulo 1 6
Introducción 6
1.1 Motivación 6
1.2 Introducción 7
1.3 Objetivos 7
1.4 Estructura del trabajo 8
Capítulo 2 9
Modelado geométrico 9
2.1 Tipos de modelado 9
2.1.1 Descomposición celular 10
2.1.2 Modelo de barridos 10
2.1.3 Geometría constructiva 10
2.1.4 Modelado por hilos 10
2.1.5 Modelo de fronteras 11
2.2 Curvas paramétricas 11
2.2.1 Polinomios de Bernstein 12
2.2.2 Curvas de Bézier 12
2.2.3 Curvas B-Spline 13

2.3 Superficies paramétricas 14
2.3.1 Superficies de Bézier 14
2.3.2 Superficies B-Spline 15
Capítulo 3 17
Método de los Momentos 17
3.1 Funciones base 19
3.1.1 Funciones base de dominio completo 19
3.1.2 Funciones base de subdominio 19
3.2 Funciones prueba 21
3.3 Matriz de acoplos 21
3.4 Método de Momentos en superficies eléctricamente grandes 21
3.5 Subdivisión de superficies NURBS 22
3.5.1 Proceso de división 22
3.6 Definición de funciones base y prueba 23
3.7 Conexión eléctrica entre dos superficies NURBS 24
3.7.1 Superficies interpolantes 24
3.7.2 Subdivisión de la superficie interpolante 25
3.8 Cercanía entre subdominios 26
3.9 Selección de la geometría 27
Capítulo 4 28
Aplicación y estructura de los datos 28
4.1 Aplicación 30
4.2 Estructura de los datos 33
4.2.1 Carpeta datos 34
4.2.2 Carpeta Guardar 39
4.2.3 Carpeta Modelos 40
4.2.4 Carpeta resul 40

4.3 Funcionamiento de la aplicación 41
4.4 Resultados 42
Capítulo 5 45
Conclusiones y líneas futuras 45
5.1 Conclusiones 45
5.2 Líneas futuras 46
Bibliografía 47
Anexo 49

Índice de figuras

Figura 2.1. Curva de Bézier y su polígono de control ……..…….. 8

Figura 2.2. Superficie NURBS y malla de control ……..…… 10

Figura 3.1. Ejemplo de funciones de dominio completo ..…… 15

Figura 3.2. Función representada mediante funciones delta ..…… 15

Figura 3.3. Función representada mediante funciones pulso ..…… 15

Figura 3.4. Función representada mediante funciones triángulo …… 15

Figura 3.5. Subdivisión de una superficie NURBS …… 18

Figura 3.6. Superficie interpolante ……..……..… 20

Figura 3.7. Subdivisión de la superficie interpolante …….. 20

Figura 4.1. Interfaz de Matlab …… 24

Figura 4.2. Interfaz gráfica de la aplicación ……..…….. 27

Figura 4.3. Fichero de datos …….. 29

Figura 4.4. Placa …… 31

Figura 4.5. Esfera …… 32

Figura 4.6. Cono inferior …… 32

Figura 4.7. Dos conos …… 31

Figura 4.8. Avión CN235 …..…… 31

Figura 4.9. Fichero de resultados …… 36

Figura 4.10. Sección radar monoestática de la esfera …….. 43

Figura 4.11. Datos para la simulación de la esfera …… 43

Figura 4.12. RCS CN235 ……... 44

Figura 4.13. Datos para la simulación del CN235 …….. 44

Capítulo 1

Introducción

1.1 Motivación

En el año 2013 comencé a estudiar el Grado en Ingeniería de las Tecnologías de Telecomunicación, en la Universidad de Cantabria. Elegí esta carrera no sólo porque tenía claro que quería estudiar una ingeniería y me encantaba el mundo de las telecomunicaciones, sino también por las grandes posibilidades de futuro que tiene esta carrera.

Avanzando cursos me di cuenta que la mención de Sistemas de Telecomunicaciones era la que mejor se adaptaba a mí, por las diferentes asignaturas que me parecían más atractivas, hasta que en cuarto asistí a la asignatura de Sistemas de Radiodeterminación, que me levantó curiosidad sobre las antenas y que podría ocurrir con ellas.

Así que esto fue lo que me motivó para elegir al tutor y el Trabajo de Fin de Grado el cual me ha aportado muchas cosas satisfactorias además de favorecer mi aprendizaje en distintos temas.

1.2 Introducción

Debido a la gran evolución que ha tenido la tecnología hasta la actualidad es importante que cuando se quiera realizar cualquier infraestructura, de lo que sea, pueda existir un buen estudio previo que nos garantice que el diseño de lo que estamos planteando sea buena y vaya a funcionar con normalidad, gracias a estos avances se ha ganado mucho en velocidad y almacenamiento en los ordenadores, lo cual es perfecto para lo que queremos plantear en este trabajo.

Más en relación con nuestro trabajo, cada vez hay más Antenas en cualquier lugar, entonces poder realizar un estudio previo, teniendo el modelo en el que se van a encontrar, de los diferentes parámetros de una antena, ver cómo van a afectar todas las estructuras que se encuentren a su alrededor, interferencias entre las antenas es muy importante, debido a que hacerlo de manera experimental es algo muy caro. Si podemos realizar simulaciones con un ordenador y que los resultados sean válidos es algo que será de gran ayuda.

Con todo esto, presentaremos una aplicación que haciendo uso de un programa de resolución de problemas electromagnéticos presente los resultados de una manera visual sencilla y que sea muy intuitiva para usuarios inexpertos que quieran hacer uso de ella, además de que usuarios expertos puedan trabajar con ella modificando más parámetros.

1.3 Objetivos

En este trabajo se persiguen los siguientes objetivos:

- Presentar una aplicación sencilla e intuitiva para el uso de usuarios inexpertos.
- Probar la evolución y por consiguiente ganancia de tiempo de cálculos, así como en los dibujos de los modelos.
- Demostrar que el programa sigue siendo muy válido y utilizable.

1.4 Estructura del trabajo

El trabajo constará de cuatro capítulos fundamentales:

- En el capítulo 2 hablaremos del modelado geométrico, presentaremos diferentes tipos de modelado y explicaremos porque hacemos uso del que al final se elige analizando sus ventajas y desventajas.
- En el capítulo 3 se presenta la técnica numérica que se utiliza en el programa del que hace uso la aplicación, dando sus características más importantes.
- El capítulo 4 constará de todas las partes que tiene nuestra aplicación, como está estructurada y que funciones tiene.
- En el capítulo 5 se lleva a cabo una conclusión del trabajo, así como unas líneas futuras.

Capítulo 2

Modelado geométrico

Elegir el modelado geométrico es una de las cosas más importantes, ya que cuanto mejor represente la realidad, mejores serán los resultados que vayamos a obtener. Para elegir como modelar debemos tener en cuenta el método numérico que vayamos a utilizar. El objetivo también es que no tengamos que modificar la geometría dependiendo de la frecuencia que vayamos a utilizar, si no que nos valga el mismo modelo para cualquier frecuencia.

2.1 Tipos de modelado

Hay muchas formas de modelar geometrías en estos momentos, pero para decidir cuál es la que más nos conviene hay algunos factores que deben tenerse en cuenta. Debemos poder modelar una cantidad de objetos muy grande, también es muy importante que se pueda hacer con la mínima información necesaria, ya que esto supondrá ganar mucho tiempo en cálculos y debe ser sencillo obtener la posición de los puntos, sus derivadas y los vectores normales. Entonces cada manera de modelar tiene sus ventajas e

inconvenientes, a continuación, presentamos algunas de las características de varios tipos de modelado.

2.1.1 Descomposición celular

La descomposición celular, combina celdas (generalmente cúbicas) las cuales se definen por su punto medio y existen 2 tipos, el primero de ellos es la enumeración de ocupación espacial en la cual las celdas son del mismo tamaño, por otro lado, el segundo tipo es el de árboles octales el cual intenta optimizar información, de manera que subdivide aquellas celdas en las que hay objeto y espacio en otras 8 nuevas celdas.

2.1.2 Modelo de barridos

El modelo de barridos hace evolucionar en el espacio una entidad geométrica la cual se llama generador y se aplica con métodos numéricos como el MM [1-2].

2.1.3 Geometría constructiva

La geometría constructiva de sólidos construye el cuerpo gracias a operaciones booleanas o transformaciones geométricas partiendo de cuerpos más sencillos (esferas, conos, cilindros, etc.).

2.1.4 Modelado por hilos

En el modelado por hilos conociendo sus dos extremos formamos la superficie y fue el primero para el que se aplicó el MM [3], tiene un gran problema y es que el mismo modelo puede representar más de un objeto.

2.1.5 Modelo de fronteras

El modelo de fronteras genera los cuerpos con los elementos geométricos que les describen y es la técnica más usada [4-6]. Se puede realizar de dos formas distintas, la primera de ellas mediante facetas planas y la segunda mediante curvas paramétricas. En el caso de hacerlo mediante facetas planas podemos representar cualquier objeto, pero el problema es que el modelo no es válido para todas las frecuencias ya que cuanta más alta sea la frecuencia tenemos que añadir más facetas. En cambio, sí modelamos mediante curvas paramétricas disminuimos la información y nos vale el mismo modelo para todas las frecuencias y lo más utilizado para modelar mediante superficies curvas son las superficies NURBS que son las que se utilizarán en el programa que usaremos. Usar superficies paramétricas nos presenta varias ventajas:

- Es bastante sencillo obtener parámetros concretos de la superficie.
- El modelo que se utiliza es muy semejante con la realidad y como dijimos con anterioridad nos vale para cualquier frecuencia.
- Las facetas planas son consideradas un caso particular de las superficies NURBS.

2.2 Curvas paramétricas

Para la construcción de estas curvas paramétricas podemos hacerlo mediante dos maneras distintas:

- Interpolación: Hace que la curva pase por todos los puntos, pero sin asegurar que dicha curva sea una curva suave.
- Aproximación: De esta manera no aseguraremos que pase por todos los puntos, pero si será una curva suave que se aproxime a los puntos y es además la forma en la que modelaremos las geometrías.

Estas curvas las construiremos a partir de curvas de Bézier que a su vez vienen definidas por los polinomios de Bernstein y a continuación daremos algunas de las características.

2.2.1 Polinomios de Bernstein

Los polinomios de Bernstein que serán la base para las curvas Bézier vienen definidas por la siguiente ecuación:

$$B_i^n(t) = \frac{n!}{i!\,(n-i)!} t^i (1-t)^{n-i} \tag{2.1}$$

2.2.2 Curvas de Bézier

Estas curvas se desarrollaron sin tener en cuenta una relación con los polinomios de Bernstein hasta que en la década de los 70 se encontró la relación, entonces una curva de Bézier se puede expresar de la siguiente manera:

$$\vec{c}(t) = \sum_{i=0}^{n} \overrightarrow{b_i} B_i^n(t) \tag{2.2}$$

Donde n será el grado de la curva y b formará un polígono de control formado por n+1 puntos. Dichas curvas tienen unas propiedades muy interesantes para el diseño de los cuerpos y por eso nos sirven de gran ayuda. En la figura 2.1 se muestra una curva de Bézier y su polígono asociado.

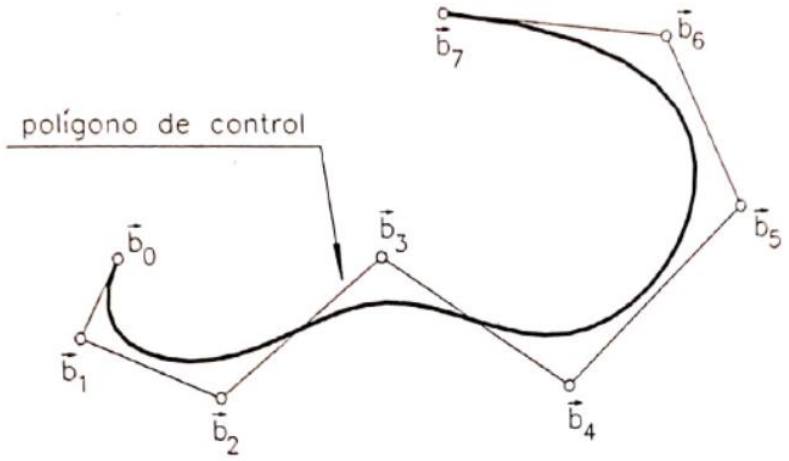

Figura 2.1 Curva de Bézier y su polígono de control

Más tarde surgió el concepto de curvas racionales [7] que lo que hacían es que en cada punto del polígono de control se le daba un peso diferente y de esta manera podías moldear la curva más a tu gusto y se podían representar algunas curvas que con las no racionales no era posible.

2.2.3 Curvas B-Spline

Para combatir los inconvenientes que tienen estas curvas, como es por ejemplo que al modificar cualquier punto de control ya se nos modificaría la curva, lo que se hace es combinar diferentes curvas de Bézier, pero de grados más bajos y estas curvas pasan a llamarse B-Spline. La cual tiene la siguiente ecuación:

$$\vec{\boldsymbol{c}}(\boldsymbol{t}) = \sum_{\boldsymbol{i=0}}^{\boldsymbol{n}} \overrightarrow{\boldsymbol{b}_i} \boldsymbol{N}_{\boldsymbol{i}}^{\boldsymbol{k}}(\boldsymbol{t}) \qquad \boldsymbol{2 \leq k \leq n+1} \tag{2.3}$$

Donde $N_i^k(t)$ son las bases B-Spline, las cuales son un conjunto de puntos que se hace llamar vector de nudos. Dependiendo de cómo sean los valores de dicho vector la curva será uniforme o no uniforme y como las curvas no uniformes nos ofrecen mayor versatilidad son las que se usarán en mayor medida. Estás nuevas curvas tienen las mismas propiedades que las curvas de Bézier, pero además se le añade que si modificamos un punto de control solo afecta localmente a esa zona de la curva.

2.3 Superficies paramétricas

En su gran mayoría, las propiedades que hemos dicho anteriormente para las curvas paramétricas son escalables a las superficies paramétricas, pero hay que añadir una variable para un barrido bidimensional.

2.3.1 Superficies de Bézier

Es por esto que podemos definir las superficies de Bézier a partir de los polinomios de Bernstein con la siguiente expresión:

$$\vec{\boldsymbol{r}}(\boldsymbol{u},\boldsymbol{v}) = \frac{\sum_{\boldsymbol{i}=\boldsymbol{0}}^{\boldsymbol{m}}\sum_{\boldsymbol{j}=\boldsymbol{0}}^{\boldsymbol{n}}\boldsymbol{w_{ij}}\vec{\boldsymbol{b}}_{\boldsymbol{ij}}\boldsymbol{B_i^m}(\boldsymbol{u})\boldsymbol{B_j^n}(\boldsymbol{v})}{\sum_{\boldsymbol{i}=\boldsymbol{0}}^{\boldsymbol{m}}\sum_{\boldsymbol{j}=\boldsymbol{0}}^{\boldsymbol{n}}\boldsymbol{w_{ij}}\boldsymbol{B_i^m}(\boldsymbol{u})\boldsymbol{B_j^n}(\boldsymbol{v})} = \frac{\boldsymbol{N}(\boldsymbol{u},\boldsymbol{v})}{\boldsymbol{D}(\boldsymbol{u},\boldsymbol{v})} \qquad (2.4)$$

En este caso en vez de tener un polígono de control lo que tenemos es una malla de control como puede verse en la figura 2.2.

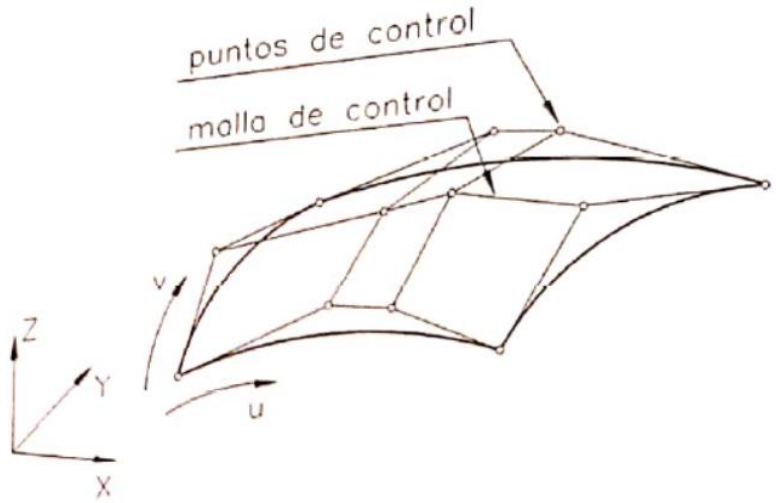

Figura 2.2 Superficie NURBS y malla de control

Es importante decir que los parámetros (u, v) tomarán valores que oscilen entre (0, 1) ya que será clave para cuando hablemos de subdividir la superficie en superficies más pequeñas ya que será mejor para aplicar el MM.

Tal y como están construidas estas superficies son derivables en todos los puntos además de que las derivadas respecto a cada parámetro (u, v) serán tangentes a las líneas isoparamétricas contrarias, lo cual será de vital importancia para cuando tengamos que definir la corriente superficial ya que nos harán falta para resolver la ecuación integral.

Como vimos con anterioridad, las curvas de Bézier tenían el problema de que al modificar un punto de control nos cambiaba la curva, en el caso de las superficies nos pasa lo mismo, así que tenemos que hacer uso de las superficies B-Spline.

2.3.2 Superficies B-Spline

Estas superficies las podemos definir como:

$$\vec{r}(u,v) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} w_{ij}\vec{d}_{ij}N_i^m(u)N_j^n(v)}{\sum_{i=0}^{m}\sum_{j=0}^{n} w_{ij}N_i^m(u)N_j^n(v)} \quad 2 \le k \le m+1; 2 \le l \le n+1 \tag{2.5}$$

Y como ya dijimos antes para las superficies es necesario definir el vector de nudos y dependiendo de cómo sean estos vectores las superficies serán uniformes o no uniformes (superficie NURBS). Para aplicar el método en el programa se utilizaron superficies de Bézier ya que son más fáciles de manejar para obtener puntos, derivadas, etc.

Capítulo 3

Método de los Momentos

El método de los Momentos es la mejor técnica para abordar el problema con estructuras comparables a la longitud de onda y para resolver problemas de scattering electromagnético, de tal manera que la solución se obtiene resolviendo un sistema de ecuaciones lineales teniendo como ventaja en este método que ya están metidos todos los efectos posibles debidos a campo directo, difracciones, reflexiones, etc.

Este será el método en el que se basa el programa de la aplicación [8-11], pero será una técnica híbrida para poder aplicarlo sobre superficies eléctricamente más grandes.

Teniendo en cuenta varios puntos sobre ecuaciones de campo eléctrico y distribuciones de corrientes podemos llegar a la Ecuación Integral de Potenciales Mixtos que sigue la siguiente expresión:

$$\vec{n} \times \vec{E}^i(\vec{r}) = \vec{n} \times [\frac{jw\mu}{4\pi} \int_S \vec{J}\left(\vec{r'}\right) G(\vec{r},\vec{r'}) ds' - \frac{1}{jw4\pi\varepsilon} \nabla \int_S \nabla \vec{J}\left(\vec{r'}\right) G(\vec{r},\vec{r'}) ds'] \quad (3.1)$$

Como cada vez es más sencillo programar técnicas en ordenadores y de esta manera obtener resultados muy buenos y con mucha mayor velocidad, las técnicas complejas o muy repetitivas se usan cada vez más y el Método de los Momentos[1-2] al final trata de resolver un sistema de ecuaciones que tiene esta forma:

$$\boldsymbol{\varphi(I) = Y} \tag{3.2}$$

Este sistema de ecuaciones no tiene una solución analítica exacta así que hay que aproximar la incógnita por una serie de funciones y llegamos a un sistema de una ecuación con N incógnitas y con una serie de coeficientes desconocidos $\alpha_n$así que para resolverlo definiremos otras funciones prueba W de lo que sacaremos un sistema de M ecuaciones con N incógnitas de la siguiente manera:

$$[\boldsymbol{Z_{mn}}][\boldsymbol{\alpha_n}] = [\boldsymbol{W_m}] \tag{3.3}$$

Y siendo $[Z_{mn}]$ una matriz no singular lo resolveremos de la siguiente manera:

$$[\boldsymbol{\alpha_n}] = [\boldsymbol{Z_{mn}}]^{-1}[\boldsymbol{W_m}] \tag{3.4}$$

3.1 Funciones base

Visto que para resolver el sistema tenemos que aproximar la incógnita a una serie de funciones, la elección de las mismas será una decisión importante ya que dependiendo de cuales se usen se ganará en diferentes cosas, bien sea en ahorro de información haciendo que la matriz tenga menos valores o haciendo que los valores de la matriz sean más fáciles de evaluar. Para hacer una primera clasificación podemos dividirlas en 2 clases: funciones base de dominio completo o de subdominio

3.1.1 Funciones base de dominio completo

Pueden tomar un valor distinto de cero en cualquier parte del dominio, esto es por ejemplo las funciones sinusoidales, pero el mayor problema es que debemos conocer la forma aproximada de la función. Figura 3.1.

3.1.2 Funciones base de subdominio

Solo toman un valor distinto de cero en una parte del dominio y lo que se hace es dividir el dominio para tener subdominios más pequeños. Las tres funciones más conocidas son:

- Funciones delta: Toman un valor distinto de cero únicamente en un punto y este será donde se encuentren dos subdominios. Figura 3.2.
- Funciones pulso: Son pulsos que tienen como altura la unidad y que están centrados en su punto medio. Figura 3.3.
- Funciones triángulo: Los triángulos están centrados en el punto en que se separar dos subdominios de al lado. Figura 3.4.

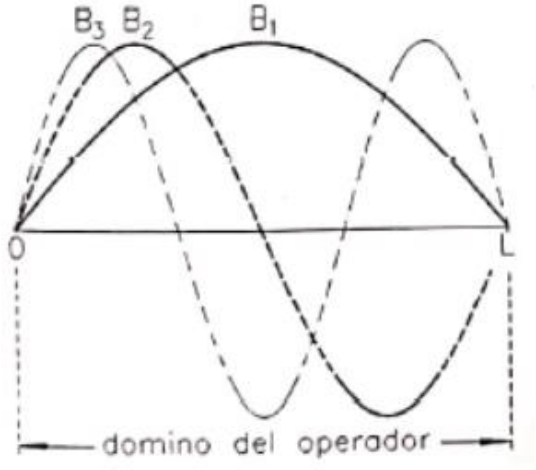

Figura 3.1 Ejemplo de funciones de dominio completo

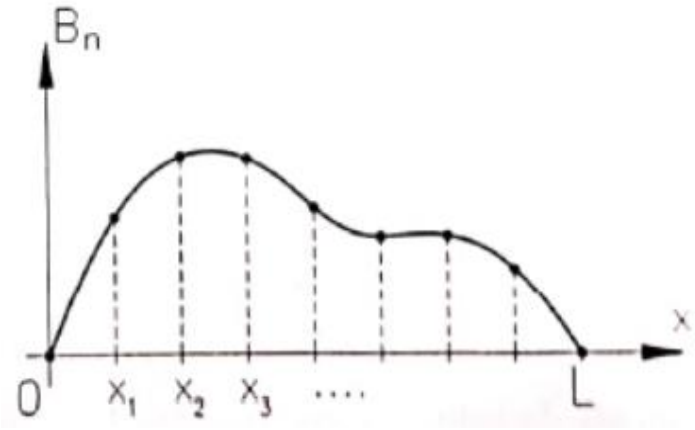

Figura 3.2 Función representada mediante funciones delta

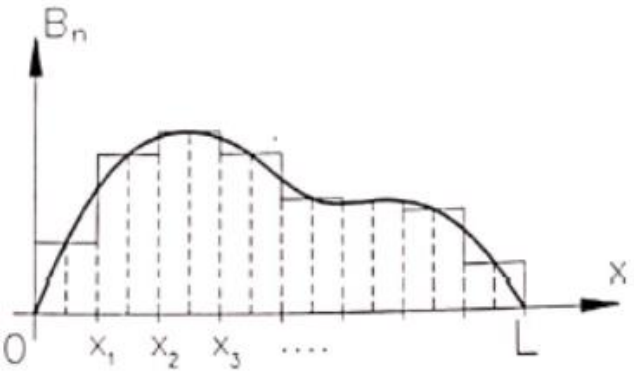

Figura 3.3 Función representada mediante funciones pulso

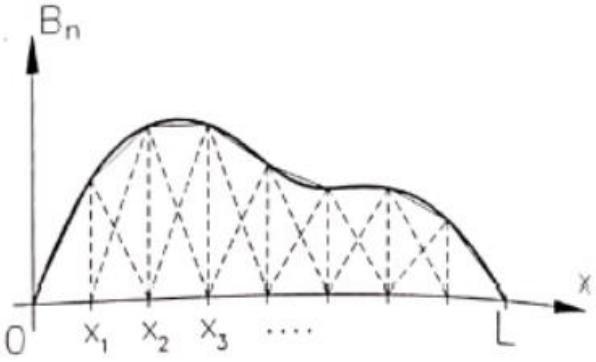

Figura 3.4 Función representada mediante funciones triángulo

Estas funciones base explicadas nos servirán para en combinación con otras y dependiendo de la geometría del problema utilizarlas como funciones base.

De estas combinaciones salen las funciones rooftop que son las más utilizadas. Por ejemplo, si sobre una placa hacemos incidir una onda plana, el comportamiento de la densidad de corriente J para J_x en la dirección x tiene la forma de función triangular y para la dirección y funciones de pulso y para J_y a la inversa, en la dirección x tiene funciones pulso y en la dirección y funciones triángulo.

3.2 Funciones prueba

Al igual que las funciones base, estas funciones también tienen en cuenta la geometría que se estudie, las funciones que se han utilizado en el programa son las funciones bladerazor (o cuchilla) y se han programa en su aproximación por tramos rectos.

3.3 Matriz de acoplos

En cualquier programa que se base en Método de Momentos, obtener esta matriz es un proceso crucial y será lo que más tiempo lleve, se trata de calcular la segunda parte de la ecuación 3.1. Dicha ecuación está compuesta por dos sumandos uno de ellos será el término inductivo y el otro el término capacitivo y se calcularán por separado.

3.4 Método de Momentos en superficies eléctricamente grandes

Para aplicar un método riguroso como lo es el Método de Momentos [12], se necesita discretizar el problema y extraer entre 8 y 10 muestras por cada longitud de onda, de esto precisamente deriva el problema de usar esta técnica con superficies eléctricamente grandes, ya que debemos guardar demasiada información.

El objetivo que persigue el programa utilizado es reducir el tamaño de la matriz de acoplos y de esta manera poder aplicar el Método de Momentos a superficies eléctricamente grandes.

Para poder aplicar este método mejor debemos subdividir las superficies y así trabajaremos de una manera más sencilla.

3.5 Subdivisión de superficies NURBS

Como hemos mencionado en el capítulo anterior se utilizaron superficies paramétricas para modelar ya que nos proporcionaba la ventaja de que el mismo modelo era completamente válido independientemente de a la frecuencia que queramos trabajar o analizar.

En el momento en que sea necesario subdividir las superficies se hará de manera individual con cada una de las superficies, primero se de analizan las curvas u=cte aproximándolas por el polígono de control, así sabemos cómo es la longitud aproximada de la curva, después se hace lo mismo para las curvas v=cte y si por ejemplo la curva u=cte tiene un tamaño como una longitud de onda se deberán hacer entre 8 y 10 subdivisiones y de igual manera para las curvas v=cte

3.5.1 Proceso de división

Los tramos en los que dividimos la curva serán de igual longitud. En la figura 3.5 tenemos una superficie NURBS limitada por cuatro curvas las cuales son demasiado grandes para poder analizarlas mediante el Método de Momentos así que hay que subdividirla.

- Primero debemos estimar la longitud de las curvas como hemos explicado en el apartado anterior y saber el número de tramos en que debemos dividir la curva.
- Seguidamente pasamos la superficie a su representación paramétrica y dividiremos las curvas en tantos tramos como hayamos visto en el primer paso.
- Por último, lo volvemos a pasar al dominio real.

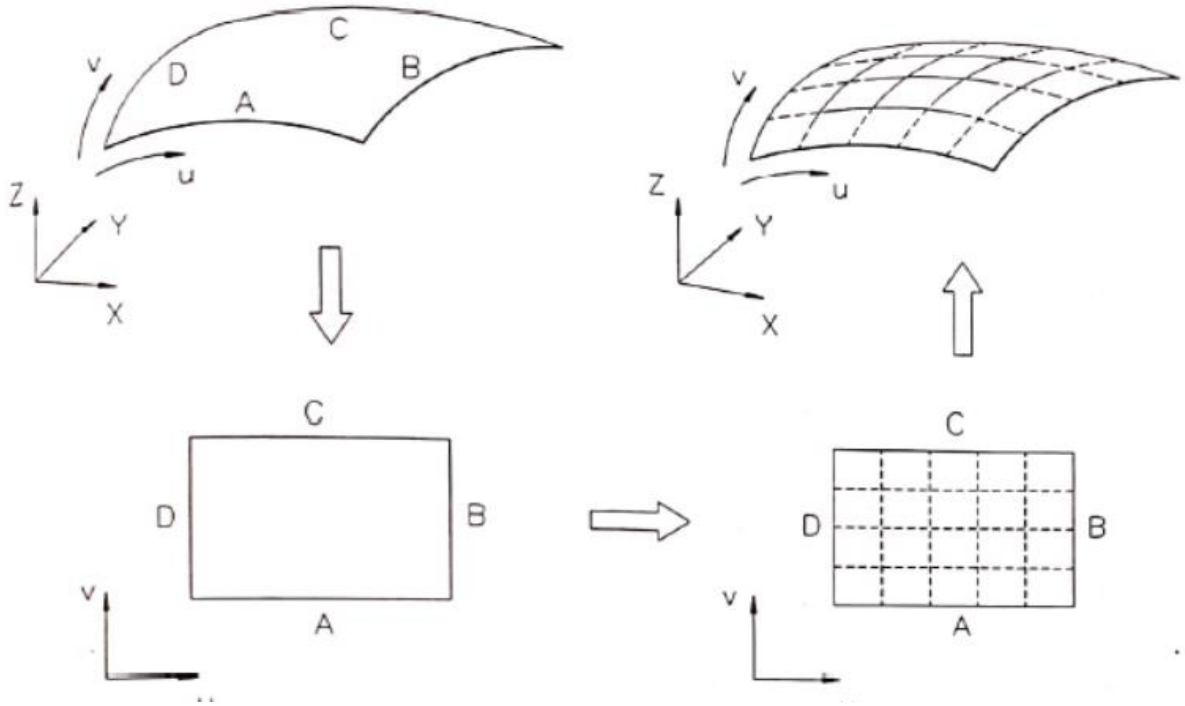

Figura 3.5 Subdivisión de una superficie NURBS

3.6 Definición de funciones base y prueba

Cuando ya tenemos la superficie dividida podemos pasar a definir las funciones base de las corrientes para el Método de Momentos.

Cada nuevo subparche viene definido por cuatro curvas y cada una tendrá su función de corriente, las obtendremos haciendo variar cada una de las coordenadas mientras dejamos la otra fija, entonces para el primer subparche la corriente será:

$$\vec{\boldsymbol{J}}(\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{J}_{\boldsymbol{i}}^{\boldsymbol{A}}(\boldsymbol{u},\boldsymbol{v})\vec{\boldsymbol{e}}_{\boldsymbol{u}}(\boldsymbol{u},\boldsymbol{v}) \qquad (3.5)$$

Siendo $\boldsymbol{J}_{\boldsymbol{i}}^{\boldsymbol{A}}$(u,v) :

$$J_i^A(u,v) = \frac{1}{S_A\sqrt{g(u,v)}} \int_{u_{min}}^{u} \sqrt{g(u,v)}\, du \tag{3.6}$$

Esto hay que hacerlo con todos los subparches y de esta manera lo tendremos para todas las de la superficie NURBS.

Como se trabaja siempre con más de una superficie NURBS, falta por decir como es en la curva que comparten dos superficies adyacentes, ya que al subdividir cada superficie de manera individual pueden surgir problemas en las conexiones eléctricas, lo que deterioraría gravemente los resultados y no los haría válidos.

3.7 Conexión eléctrica entre dos superficies NURBS

Normalmente todas las geometrías que se trabajen van a venir definidas por más de una superficie NURBS ya que por ejemplo se podría definir un cilindro con una superficie únicamente, pero al hacerlo con más conseguiremos optimizar los cálculos haciéndolos más sencillos de manejar, siempre y cuando no pongamos superficies de más, hay que buscar el equilibrio.

Se deben definir las funciones base donde se unen dos superficies.

3.7.1 Superficies interpolantes

En la unión de dos superficies NURBS, las cuales están subdivididas, los subparches no tienen por qué coincidir ya que no tienen por qué tener el mismo número de divisiones y se debe crear una nueva superficie, la cual se llamará superficie interpolante como se ve en la figura 3.6.

Lo que hay que hacer es conseguir una nube de puntos situada entre las dos superficies y de ahí sacar una nueva superficie NURBS mediante

interpolación, esta nueva superficie tendrá un tamaño grande para aplicar el Método de Momentos así que debemos subdividirla.

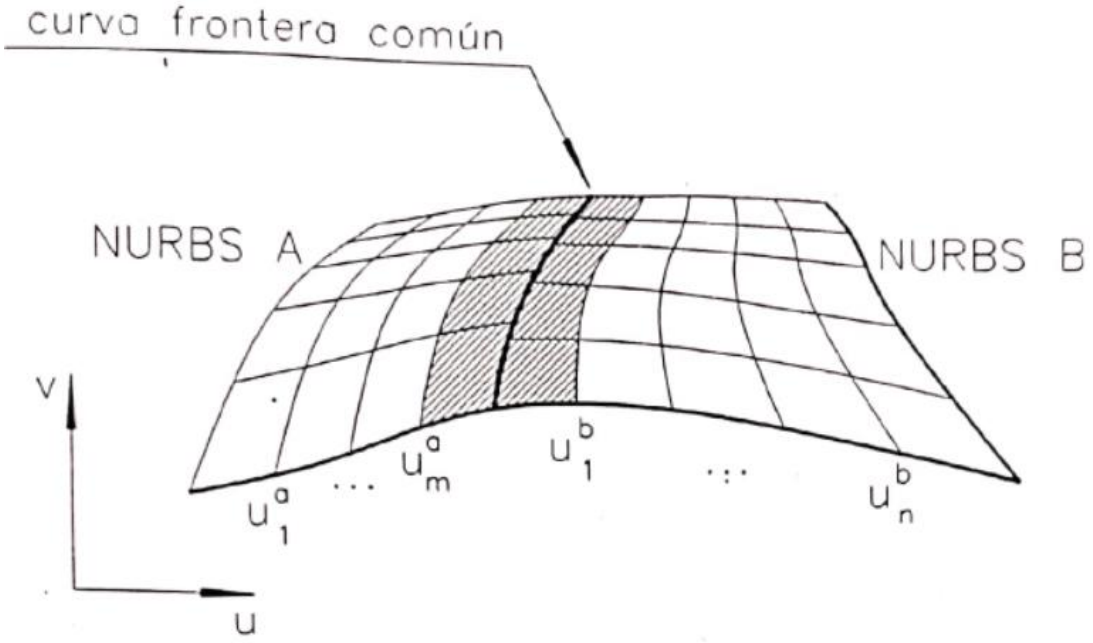

Figura 3.6 Superficie interpolante

3.7.2 Subdivisión de la superficie interpolante

Esta nueva superficie tendrá las mismas propiedades que cualquier superficie NURBS, así que la dividiremos realizando lo mismo que hacíamos anteriormente como se puede ver en la figura 3.7.

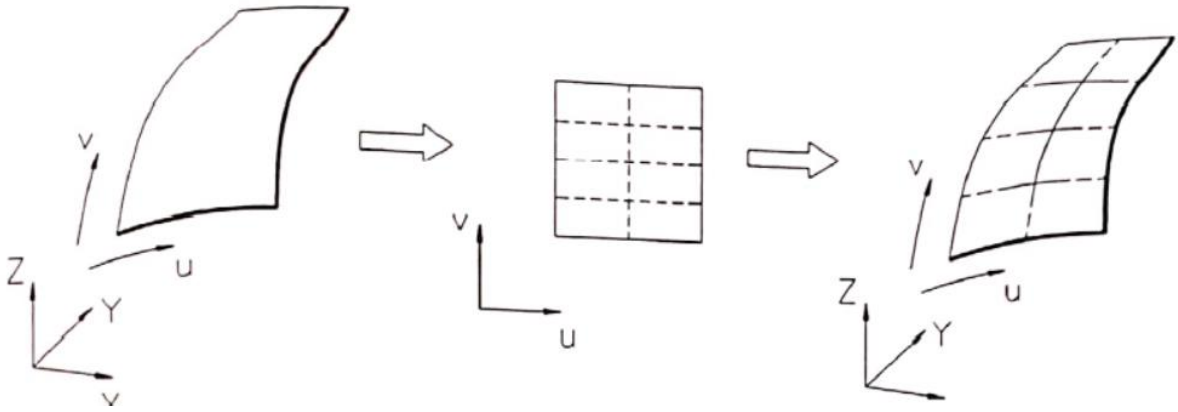

Figura 3.7 Subdivisión de la superficie interpolante

Finalmente debemos colocarla donde corresponde en el modelo real.

Esto se hace para hacer más sencilla la geometría que vamos a utilizar. Llegados a este punto los que queda por mejorar es en tiempos de cálculo con superficies grandes.

3.8 Cercanía entre subdominios

Anteriormente hemos hablado sobre la matriz de acoplos y como vimos en la ecuación 3.1 estaba compuesta por un término capacitivo y otro inductivo, pues el cálculo de uno de esos términos depende de diferentes factores, los dos más importantes son:

- La distancia entre subdominios implicados en el mismo cálculo.
- Densidad de corriente que genera cada uno de los subdominios.

El acoplo entre dos subdominios tiene mucho que ver con la distancia que los separa y viene definido por la siguiente ecuación:

$$\boldsymbol{G}(\vec{\boldsymbol{r}}, \vec{\boldsymbol{r}}') = \frac{\boldsymbol{e}^{-\boldsymbol{jk}|\vec{\boldsymbol{r}}-\vec{\boldsymbol{r}}'|}}{|\vec{\boldsymbol{r}} - \vec{\boldsymbol{r}}'|} \qquad (3.7)$$

Cuanta más distancia haya entre dos puntos menor será el acoplo entre ellos, esto es de vital importancia para las superficies eléctricamente grandes ya que puede que haya valores que no nos aporten nada y podamos despreciarlos.

A partir de esta idea se define un parámetro que es la máxima distancia de acoplo entre subdominios.

De esta manera todos los subdominios que estén fuera de esa distancia máxima no serán considerados en el cálculo lo que supone un ahorro de información bastante grande.

En la parte que corresponde a la densidad de corriente que generan los subdominios es importante decir que habrá unos subdominios que hay que tener en cuenta siempre por el efecto de las corrientes independientemente de la distancia que los separe.

3.9 Selección de la geometría

Otra de las aproximaciones que tiene aplicadas el programa es seleccionar la parte de la geometría que se vaya a analizar, ya que al ser superficies tan grandes habrá partes que no influyan en el resultado final así que si no las tenemos en cuenta ahorraremos información y tiempos de cálculo, esto se hará mediante técnicas de trazado de rayos para saber cuáles son las partes visibles de la geometría.

Capítulo 4

Aplicación y estructura de los datos

Para el diseño de esta aplicación se puede utilizar diferente software, en este caso se ha elegido Matlab, ya que es un programa que se ha utilizado durante la carrera y que existen suficientes conocimientos previos como para desenvolvernos con facilidad con dicho software.

Más concretamente basaremos la aplicación en una GUI, es decir, una interfaz gráfica de usuario la cual permite una utilización muy sencilla del software desarrollado sin necesidad de conocer el lenguaje en que esté programado.

Matlab facilita esto ya que posee una herramienta de creación de GUI's [13] que genera la interfaz básica y facilita su posterior modificación añadiendo automáticamente parte del código necesario. El entorno de programación puede verse en la figura 4.1.

Una vez se tenga esta interfaz se pueden añadir diferentes componentes, tales como los que se muestran en la siguiente tabla:

Control	Valor de estilo	Descripción
Check box	‘checkbox’	Indica el estado de una opción o atributo
Editable Text	‘edit’	Caja para editar texto
Pop-up menú	‘popupmenu’	Lista de opciones
List Box	‘listbox’	Lista deslizable
Push Button	‘pushbutton’	Genera un evento al ser pulsado
Radio Button	‘radiobutton’	Indica una opción que se puede seleccionar
Toggle Button	‘togglebutton’	Tiene como estados ‘on’ o ‘off’
Slider	‘slider’	Se usa para representar un rango de valores
Static Text	‘text’	Muestra un texto en una caja
Panel Button	‘panel’	Agrupa varios botones
Button Group	‘buttongroup’	Permite exclusividad de selección con los botones

Tabla 4.1 Distintas opciones para añadir en la GUI

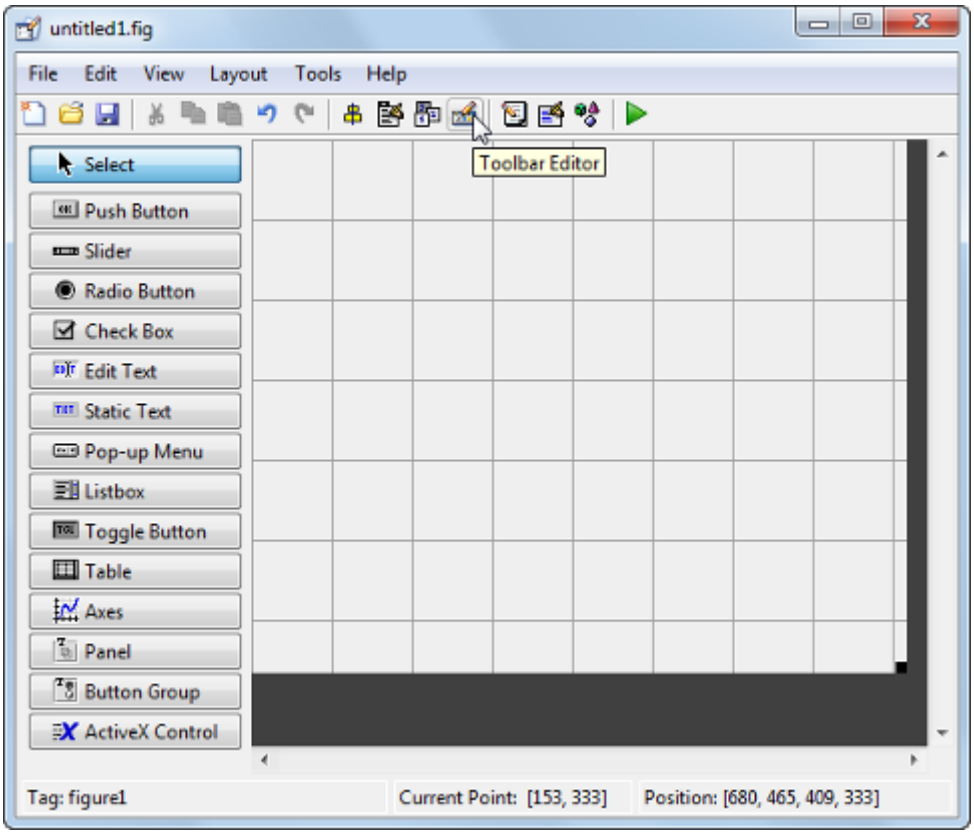

Figura 4.1 Interfaz de Matlab

Una vez hemos ubicado todos los elementos que queremos, si hacemos click derecho sobre uno de ellos se nos abre un pequeño menú, donde la opción más importante será la de 'View Callbacks'. Matlab generará las líneas de código correspondientes que serán la subrutina que se ejecute cuando se interaccione con dicho objeto, que permanecerá inactivo hasta que el usuario genere un evento tal como pulsar un botón, escribir un texto, etc. En ese apartado es donde escribiremos las líneas de código que hagan lo que queremos que realice el programa, un ejemplo sencillo sería poner dos huecos para texto con un operador '+' en medio y poner un botón de calcular, de esta manera el usuario rellenaría los sumandos y al pulsar el botón 'calcular' le daría el resultado.

Así que como podemos observar es una herramienta muy útil para hacer cosas muy sencillas para el usuario, de tal manera que el que vaya a utilizar la aplicación no tenga por qué tener ningún tipo de conocimiento sobre programación.

Una vez hayamos creado nuestro diseño, cuando lo queramos guardar, Matlab nos generará dos archivos distintos, ambos con el nombre de la GUI, uno de ellos será .m y el otro .fig. En el archivo .m estarán todas las líneas de código y subrutinas que se vayan a ejecutar y el archivo .fig será el archivo que nos permita modificar la interfaz de usuario, bien sea modificando la posición de los elementos o añadiendo o quitando elementos.

Además, es una herramienta muy buena para trabajar con ficheros y ejecutables externos como es nuestro caso, así que es una herramienta ideal para lo que se quiere presentar en este trabajo.

Ahora vamos a describir como tenemos organizada nuestra GUI, que elementos la componen y el porqué de esos elementos.

4.1 Aplicación

Como hemos dicho anteriormente, lo que buscamos con esta aplicación es que sea algo muy sencillo para el usuario, que simplemente tenga que

limitarse a introducir los valores necesarios. Pensando en esto, habrá parámetros que propondrán un valor por defecto, ya que serán los menos obvios para modificar, dejando el hueco libre de los parámetros que sean más fáciles para cambiar. Igualmente, los parámetros que tengan valor por defecto también serán modificables, de esta manera si un usuario más experto hace uso de la aplicación, pueda confeccionarlo todo a su gusto.

Dicho esto, la interfaz de nuestra aplicación será la que se ve en la figura 4.2, la cual estará compuesta por:

- Un botón de cargar el cual al pulsarle nos llevará a los elementos que hayamos guardado previamente para cargarlos y no tener que volver a simularlo.
- Un menú desplegable en el que seleccionaremos lo que queremos calcular.
- Otro menú desplegable en paralelo en el cual elegiremos el modelo sobre el que queremos estudiar.
- Un panel que aparece al seleccionar lo que queremos calcular en el que en su interior aparecen todos los datos necesarios para su cálculo.
- Los ángulos θ y Φ.
- La frecuencia a la cual vamos a analizar.
- El máximo número de iteraciones.
- El máximo error permitido.
- Divisiones por longitud de onda de los parches curvos y planos.
- El tipo de sección radar.
- El factor de normalización del campo.
- Los cortes en Theta y Phi constantes.
- El tipo de cálculo si es visible o completo.
- La Theta y Phi iniciales, así como el paso y el número de muestras.
- Un botón de ejecutar para dar paso a la ejecución.

- Un botón de guardar, en el que podremos guardar las simulaciones.

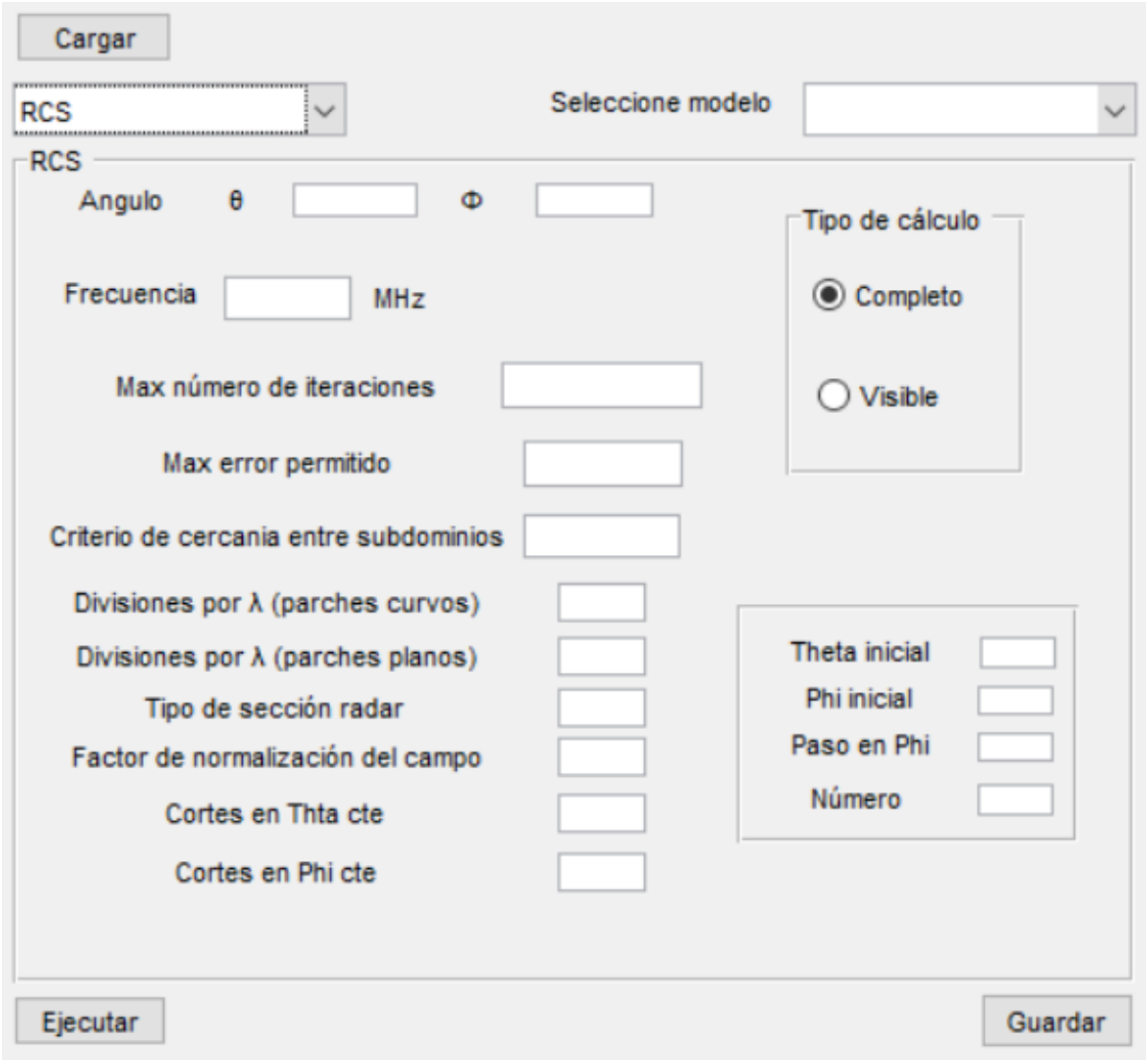

Figura 4.2 Interfaz gráfica de la aplicación

Para enlazar con los capítulos anteriores, vamos a decir en los casos más importantes donde está la relación con lo explicado.

Por ejemplo, el parámetro “criterio de cercanía entre subdominios” está directamente relacionado con lo que comentamos en el capítulo 3. Es decir, cuando un subdominio está muy separado de otro, es posible que su acoplo pueda ser despreciable de modo que se reduzca fuertemente el tiempo de

cálculo y la ocupación de la matriz de impedancias sin que eso signifique un empeoramiento de los resultados obtenidos.

Cuando hablamos de las “divisiones por longitud de onda” que se piden la aplicación es debido al proceso de división de las superficies NURBS, de modo que cuanto más alto sea este número (siempre moviéndose en el intervalo [6,10]), mayor precisión obtendremos en los cálculos, pero también mayor será el tiempo de cálculo, ya que cuantas más divisiones se hagan se obtendrán más superficies y todo el proceso se alarga. Como ya dijimos en el capítulo 2, un valor de 8 divisiones por cada longitud de onda es lo más habitual.

En el cuadro de “tipo de cálculo”, tenemos dos opciones, completo y visible. En este cuadro se hace referencia a los explicado en el capítulo 3 en el apartado de la selección de la geometría, elegiremos si se hace en torno a la geometría completa o solo en la parte visible desde la antena, esto tiene sentido para superficies eléctricamente grandes, por ejemplo si queremos analizar un avión, dependiendo de donde esté situada la antena será mejor hacerlo solo de la parte visible ya que habrá zonas que no le afecten simplemente por visibilidad.

En la aplicación también se pide un “valor máximo de iteraciones” así como un “error máximo”. Esto está relacionado con el Método de los Momentos en el capítulo 3, cuando hablamos de cómo resolver la ecuación 3.7, ya que para obtener esa matriz inversa se utiliza un método iterativo porque no se puede obtener de manera exacta, entonces en el cuadro de número de iteraciones se pone cuantas queremos que sea el máximo que haga y en el máximo error permitido el error que queremos que tenga, dicho esto puede ser que habiendo hecho todo el número de iteraciones no hayamos conseguido llegar al error y esté por encima, pero puede ser un proceso demasiado largo, por eso están ambos máximos, para que corte o bien por un lado o bien por otro.

4.2 Estructura de los datos

Para estructurarlo de manera que sea más sencillo de programar, donde tengamos la interfaz gráfica crearemos distintas carpetas, las cuales describiremos ahora.

Existirá una carpeta que se llamará ‘datos’, otra ‘Guardar’, otra ‘Modelos’ y otra que se llamará ‘resul’.

Explicaremos también como se generar los ficheros, el por qué y si tenemos que hacer uso de alguna subrutina aparte.

4.2.1 Carpeta datos

Esta carpeta constará de 2 ficheros, uno de ellos será el fichero ‘datos’ y el otro de ellos será el fichero ‘geo’.

El fichero ´datos’ tendrá la apariencia que se muestra en la figura 4.3.

```
RCS       ! Tipo de calculo a realizar: RCS, RCM, DIA, ACO, DIP o ACP.
90.0,0.0  ! Angulos THETA,PHI.
1.000E+9  ! Frecuencia.
0,0.0     ! Metodo hibrido: 0.- completo, 1.- visible. Distancia cercano.
5000      ! Numero maximo de iteraciones.
1.0E-03   ! Maximo error permitido.
200.0     ! Criterio de cercania entre subdominios.
8         ! Divisiones por lambda (parches curvos).
8         ! Divisiones por lambda (parches planos).
A         ! Tipo de seccion radar o patron de radiacion.
1.        ! Factor de normalizacion del campo.
0         ! Cortes en theta constante
1         ! Cortes en phi constante
90.0,0.0,1.0,361  ! Theta, Phi inicial, Paso en Phi, Numero de muestras.
```

Figura 4.3 Fichero de datos

El fichero de datos viene estructurado de la siguiente manera:

- En la primera línea se recoge el campo del tipo de cálculo a realizar, hay 6 tipos disponibles en este apartado RCS, RCM, DIA, ACO, DIP y ACP.
- La segunda línea tendrá los ángulos Theta y Phi.
- La tercera línea será la frecuencia a la cual queremos analizar el modelo, en la aplicación se pide en MHz.

- La cuarta línea dirá cuál es el método utilizado, puede ser o completo o visible, además vendrá acompañado de un número que en el caso del completo siempre será 0 y si la elección es visible tendrá la distancia medida en longitudes de onda hasta donde queremos que afecten a los resultados.
- En la quinta línea tendremos el número máximo de iteraciones.
- En la sexta línea estará el máximo error que queremos admitir.
- La sexta línea será ocupada por el criterio de cercanía entre subdominios.
- En la séptima y octava línea estarán las divisiones por cada longitud de onda de los parches curvos y planos respectivamente.
- En la novena línea estará el tipo de sección radar o patrón de radiación en el que la mayoría de los casos será de tipo A.
- La décima línea será ocupada por el factor de normalización del campo.
- En las líneas número once y doce tendremos los cortes en Theta y Phi constantes respectivamente.
- Por último, en la línea trece estará la Theta y Phi iniciales, así como el paso en Phi y el número de muestras que queremos.

En esta aplicación tendremos disponibles todos los cálculos mencionados en la línea 1, pero a la hora de simular nos hemos centrado en la RCS para probar que sigue siendo válido, ya que para algún tipo de cálculo es necesario otro fichero con otros datos.

En el caso del fichero ‘geo’ lo que tiene es la geometría que se va a analizar, en este caso los ficheros son bastante complicados, ya que lo que hay en su interior es lo necesario para que podamos dibujar los modelos en Matlab, así que en vez de mostrar como es el fichero en las siguientes figuras se verán los modelos que hay para analizar en la actualidad.

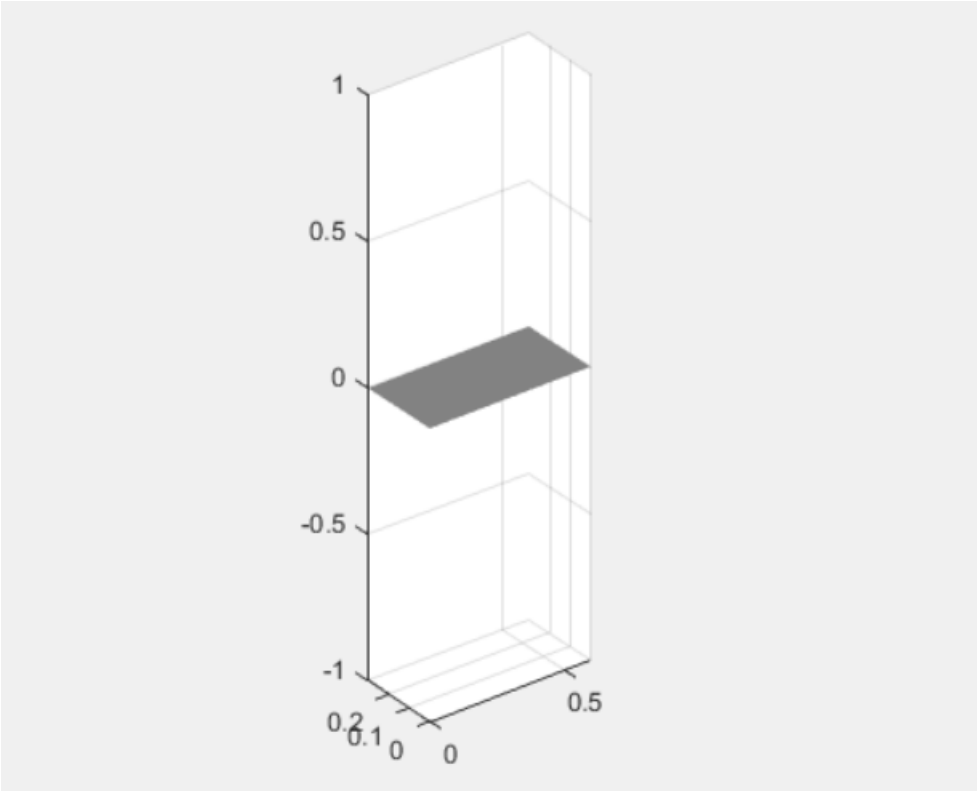

Figura 4.4 Placa

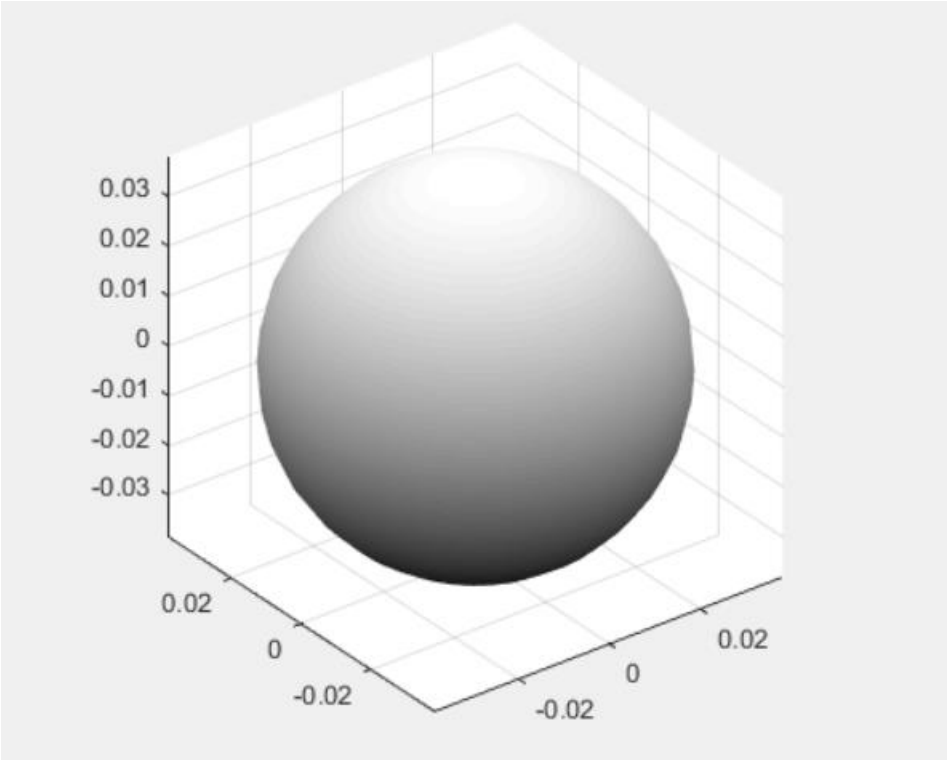

Figura 4.5 Esfera

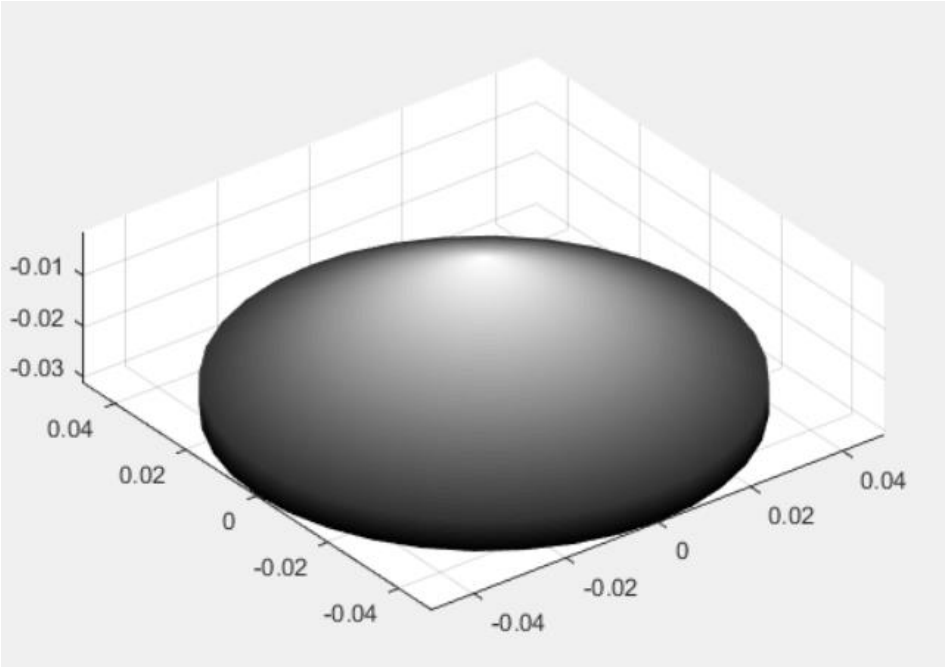

Figura 4.6 Cono Inferior

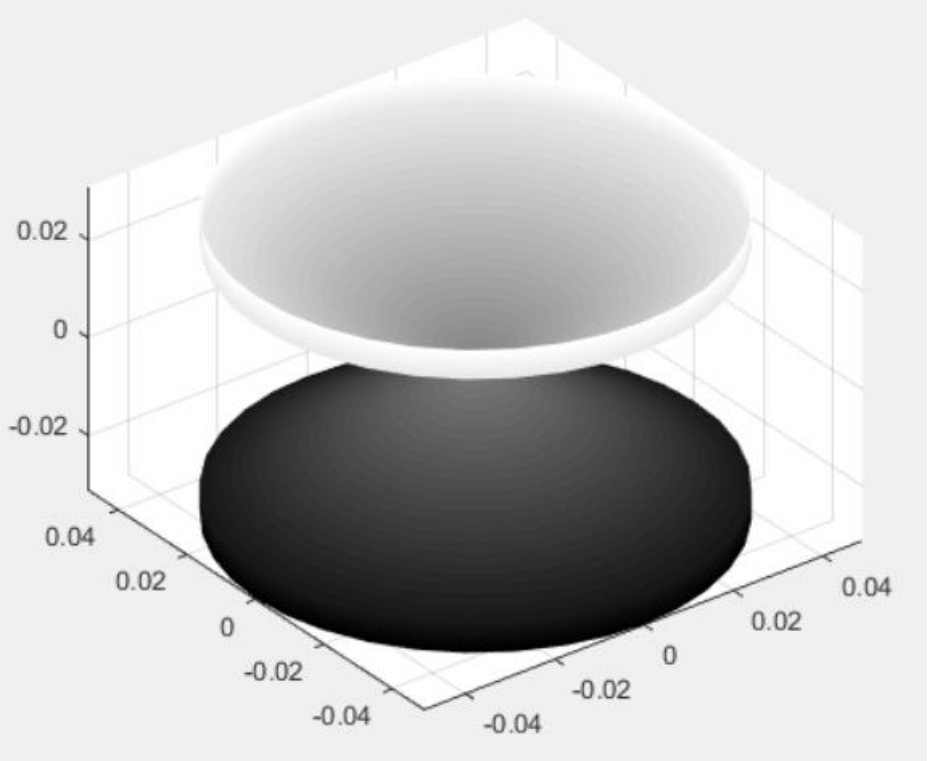

Figura 4.7 Dos Conos

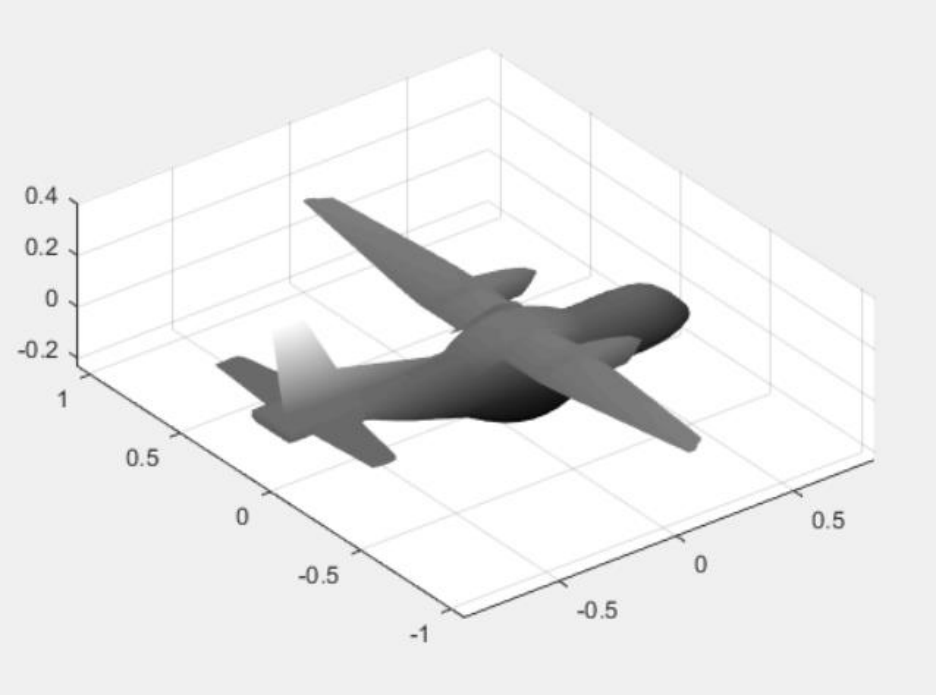

Figura 4.8 Réplica avión CN235

El programa que se usa en la aplicación necesita que todo esté en su sitio y con el nombre que tiene desde el principio ya que, si no, no detecta los archivos

bien y no nos llega a dar el resultado, entonces para programar la parte de los dibujos lo que hacemos es mover la opción que han elegido y cambiarle el nombre en esa carpeta como puede verse en los anexos de código.

4.2.2 Carpeta Guardar

En esta carpeta será donde guardemos los archivos de las simulaciones, siempre y cuando el usuario quiera guardar los datos. Esta carpeta está creada para que sea todo más cómodo a la hora de programar y también para tener todo ordenado de cara a simulaciones que se hagan, de esta manera lo que haremos será que cuando se pulse el botón de 'Guardar' la aplicación nos abrirá el típico menú para guardar y el usuario deberá introducir el nombre que quiere para las simulaciones. Entonces lo que el programa hará internamente será guardar 3 archivos con el mismo nombre, pero variando la extensión, si por ejemplo hemos dado el nombre de 'Prueba' en la carpeta Guardar aparecerán tres archivos distintos, 'Prueba.dat', 'Prueba.geo' y 'Prueba.resul'. En el primero estarán los datos introducidos en la aplicación, en el segundo estará la geometría en la que han sido analizados y en el tercero y último será el fichero de resultados.

Además, tenerlo organizado de esta manera nos da mucha facilidad para cuando se quieran cargar los resultados, ya que cuando el usuario pulse el botón 'Cargar' le abrirá directamente esta carpeta y tan solo tendrá que seleccionar el archivo que desee volver a ver, lo cual le permitirá visualizar los resultados de nuevo y también modificar el fichero de datos cambiando valores en la aplicación.

4.2.3 Carpeta Modelos

Esta carpeta está creada también con una finalidad muy similar a la anterior y es para tener los modelos ordenados de esta manera si queremos añadir más modelos el cambio en la aplicación es muy sencillo, ya que solo habría que añadir una opción en el menú y copiar y pegar unas líneas de código como puede verse en el anexo.

Cuando sale el menú desplegable del modelo y el usuario elige una de las opciones que se le dan, el programa copia el archivo y le da el nombre de geo, además de introducirlo en la carpeta de datos.

No existe ningún problema con seleccionar 2 geometrías, ya que la última que se seleccione será con la que el programa calcule los resultados, pueden visualizarse sin ningún problema.

4.2.4 Carpeta resul

Esta carpeta es la carpeta de los resultados, la cual únicamente contiene un fichero llamado ‘resul’ el cual tiene el aspecto que se muestra en la figura 4.9. En este fichero están los resultados de la simulación, el cual leeremos con una subrutina aparte a la cual llamaremos al pulsar el botón ejecutar, dicha subrutina lee el fichero quitando las dos primeras líneas ya que son únicamente comentarios, las almacena en variables y representamos la RCS en función de la Theta, como dijimos anteriormente nos centramos en la RCS y dibuja el resultado, el código que hace esto se encuentra en el anexo.

TETA(GD)	FI (GD)	ETHETA	EPHI	SECRAN(RS)
********	*******	****dB****	****dB****	****dB****
0.00	90.00	-.1260E+03	-.2818E+02	-.2818E+02
1.00	90.00	-.5419E+02	-.2818E+02	-.2817E+02
2.00	90.00	-.4817E+02	-.2819E+02	-.2814E+02
3.00	90.00	-.4465E+02	-.2819E+02	-.2810E+02
4.00	90.00	-.4215E+02	-.2820E+02	-.2803E+02
5.00	90.00	-.4022E+02	-.2822E+02	-.2795E+02
6.00	90.00	-.3864E+02	-.2823E+02	-.2785E+02
7.00	90.00	-.3730E+02	-.2825E+02	-.2774E+02
8.00	90.00	-.3615E+02	-.2827E+02	-.2761E+02
9.00	90.00	-.3513E+02	-.2829E+02	-.2747E+02
10.00	90.00	-.3422E+02	-.2832E+02	-.2732E+02
11.00	90.00	-.3340E+02	-.2835E+02	-.2717E+02
12.00	90.00	-.3265E+02	-.2838E+02	-.2700E+02
13.00	90.00	-.3197E+02	-.2841E+02	-.2682E+02
14.00	90.00	-.3133E+02	-.2845E+02	-.2665E+02
15.00	90.00	-.3074E+02	-.2849E+02	-.2646E+02
16.00	90.00	-.3019E+02	-.2853E+02	-.2627E+02
17.00	90.00	-.2968E+02	-.2858E+02	-.2608E+02
18.00	90.00	-.2919E+02	-.2863E+02	-.2589E+02
19.00	90.00	-.2874E+02	-.2868E+02	-.2570E+02
20.00	90.00	-.2831E+02	-.2873E+02	-.2550E+02
21.00	90.00	-.2790E+02	-.2879E+02	-.2531E+02
22.00	90.00	-.2751E+02	-.2885E+02	-.2512E+02
23.00	90.00	-.2714E+02	-.2891E+02	-.2493E+02
24.00	90.00	-.2679E+02	-.2898E+02	-.2474E+02
25.00	90.00	-.2645E+02	-.2905E+02	-.2455E+02
26.00	90.00	-.2613E+02	-.2912E+02	-.2436E+02

Figura 4.9 Fichero de resultados

Como puede apreciarse en la imagen, solo llega hasta el 26.00 pero es porque está recortada, tendrá tantas líneas como número de muestras hayamos puesto en la aplicación, dichas muestras avanzarán también en concordancia con el paso que hayamos puesto.

4.3 Funcionamiento de la aplicación

Una vez que se abre la aplicación aparece con los dos menús y los tres botones principales, sin nada que rellenar, se debe escoger un tipo de cálculo en el primero de los menús para que aparezca lo que hay que rellenar, en algunos casos pondremos valores por defecto, ya que como hemos dicho uno de los objetivos es hacerlo muy sencillo para el usuario, entonces los parámetros que consideramos más complejos tendrán unos valores por defecto, el resto los

rellenará el usuario. Estos parámetros que vienen por defecto son también modificables, si un usuario con más nivel usa esta aplicación, podrá modificar todos los parámetros que desee.

Una vez tengamos todos los valores puestos podemos seleccionar la geometría, abrimos el menú y elegimos la que queramos analizar, el programa abrirá otra ventana en la que dibujará la figura.

Cuando tengamos todo relleno y seleccionado podemos proceder a pulsar el botón ejecutar, este botón lo que hace es rellenar el fichero de datos y llamar a la subrutina que lee y dibuja los resultados, obteniendo en otra ventana el resultado final.

Todo el código que se ha utilizado se puede ver en el anexo.

4.4 Resultados

Para finalizar este capítulo se presentan dos resultados de dos simulaciones, la primera de ellas para probar que el programa funciona correctamente y que por tanto puede seguir siendo usado, para demostrarlo presentaremos la simulación de la sección radar monoestática de una esfera figura 4.10. La otra simulación que se presenta será con el avión CN235, se ha analizado la RCS con el método completo figura 4.12 lo cual nunca se había hecho con este programa y ha tardado 15 horas con un ordenador de características normales, así que hay una gran ganancia en tiempos de cálculo debido al avance de la tecnología. En la figura 4.11 y la figura 4.13 se muestran los ficheros de datos con los que se han simulado.

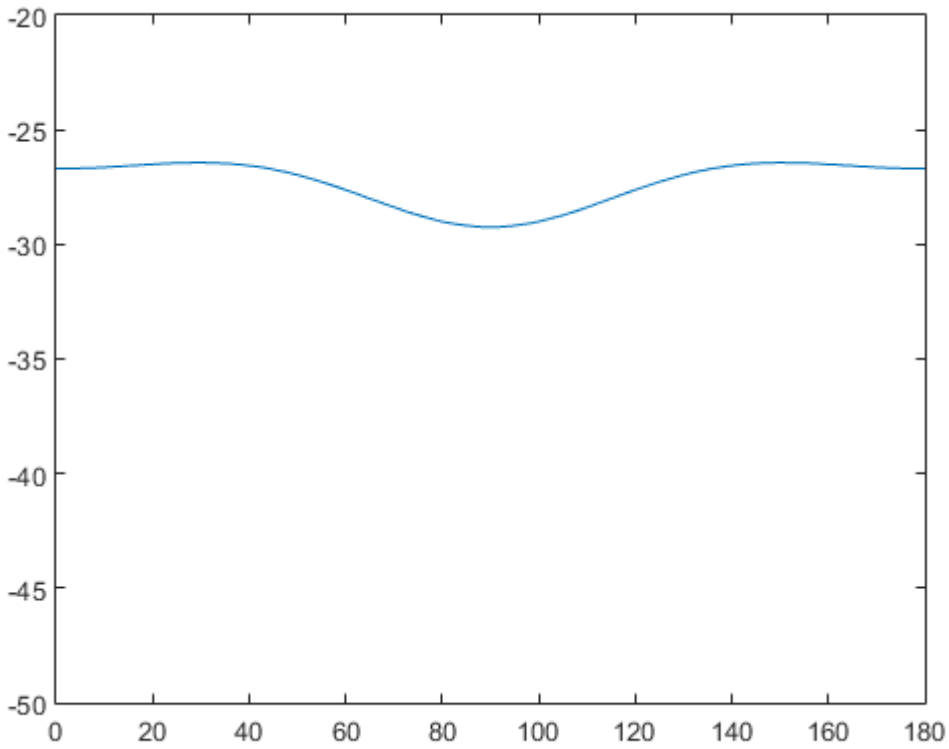

Figura 4.10 Sección radar monoestática de la esfera

```
RCM      ! Tipo de calculo a realizar: RCS, RCM, DIA, ACO, DIP o ACP.
0.0,0.0  ! Angulos THETA,PHI.
2.000E+9 ! Frecuencia.
0,0.0    ! Metodo hibrido: 0.- completo, 1.- visible. Distancia cercano.
5000     ! Numero maximo de iteraciones.
1.0E-03  ! Maximo error permitido.
200.0    ! Criterio de cercania entre subdominios.
8        ! Divisiones por lambda (parches curvos).
8        ! Divisiones por lambda (parches planos).
A        ! Tipo de seccion radar o patron de radiacion.
1.       ! Factor de normalizacion del campo.
0        ! Cortes en theta constante
1        ! Cortes en phi constante
90.0,0.0,1.0,361  ! Theta, Phi inicial, Paso en Phi, Numero de muestras.
```

Figura 4.11 Datos para la simulación de la esfera

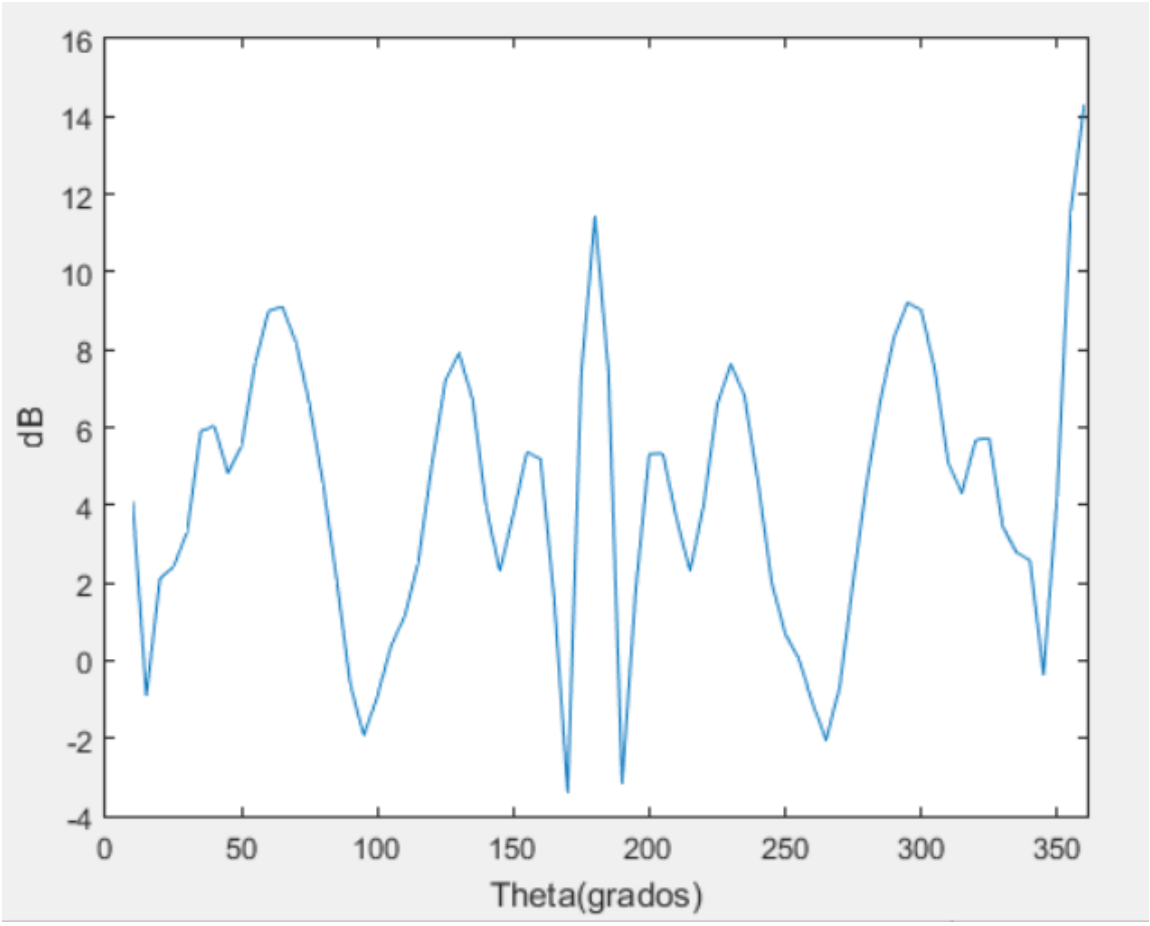

Figura 4.12 RCS CN235

```
RCM      ! Tipo de calculo a realizar: RCS, RCM, DIA, ACO, DIP o ACP.
0.0,0.0  ! Angulos THETA,PHI.
0.500E+9 ! Frecuencia.
0,0.0    ! Metodo hibrido: 0.- completo, 1.- visible. Distancia cercano.
5000     ! Numero maximo de iteraciones.
5.0E-03  ! Maximo error permitido.
200.0    ! Criterio de cercania entre subdominios.
8        ! Divisiones por lambda (parches curvos).
8        ! Divisiones por lambda (parches planos).
A        ! Tipo de seccion radar o patron de radiacion.
1.       ! Factor de normalizacion del campo.
0        ! Cortes en theta constante
1        ! Cortes en phi constante
90.0,0.0,5.0,73  ! Theta, Phi inicial, Paso en Phi, Numero de muestras.
```

Figura 4.13 Datos para la simulación del CN235

Capítulo 5

Conclusiones y líneas futuras

5.1 Conclusiones

Lo que se buscaba en este trabajo principalmente es ver que el programa que hemos utilizado sigue siendo muy válido para analizar estructuras eléctricamente grandes y hemos visto que esto es así. Es un programa basado en el Método de los Momentos, pero combinándolo con técnicas propias de la alta frecuencia, así lo que conseguimos es reducir el tamaño de la matriz de impedancias con la consecuencia de que ganamos en tiempos de cálculo.

También vemos que se gana en los dibujos de los modelos, ya que antes el programa generaba ficheros de AutoCad, esto conllevaba que era más difícil de dibujar, pues que se hacía renderizando y tardaba muchísimo más de que tarda ahora. No hay que hacer uso de un programa externo para compilar esos archivos y dibujarlos, si no que pasando los ficheros a Matlab podemos dibujarlos desde este mismo software y de una manera mucho más rápida.

Otro de los objetivos que perseguíamos era que la aplicación que se desarrolla en este trabajo fuese una aplicación muy sencilla y que pudiese utilizar cualquier usuario, sin necesidad de tener conceptos previos de programación, ni de modelado, ni tampoco de las técnicas numéricas que se utilizan para llegar al resultado.

Vista la interface que presenta el programa es una aplicación muy sencilla de trabajar y para gente que este en los primeros aprendizajes sobre RCS viene muy bien para poder ver los resultados con las gráficas, además de que es muy rápido.

5.2 Líneas futuras

Aunque en el trabajo solo se presenten 5 modelos sobre los que poder realizar simulaciones, una de las cosas que se puede hacer en el futuro es añadir más modelos, cualquier cosa que se quiera analizar puede modelarse de la misma manera que se hace en el programa que utiliza esta aplicación y añadirlo de manera muy sencilla a nuestra aplicación, ya que solo habría que añadir el fichero a la carpeta correspondiente y añadir un par de líneas de código que son simplemente copiar y pegar.

También se ofrecen diferentes tipos de cálculo en los que no hemos entrado, ya que como hemos dicho anteriormente solo nos hemos centrado en la RCS, bien hay varios tipos de cálculo que lo que necesitan es un fichero externo de una antena, pero necesita la posición exacta de la antena en el objeto, ya que si esta antena la colocamos en un sitio en el que en la realidad no va a estar ahí no valdrá de nada el estudio que hagamos, como no es objeto de este trabajo sacar la posición exacta en la que se colocaría la antena, ya que no es un proceso precisamente sencillo, lo dejamos como una ventana abierta para que se pueda trabajar sobre ello.

Otra de las cosas sobre las que se podría trabajar es la generalización del programa que se utiliza en esta aplicación, como hemos comentado necesita que los archivos estén siempre en lugares concretos y con nombres concretos, si se generalizase y simplemente tuviese diferentes carpetas donde elegir el modelos y si pudiese visualizar sería más cómodo, de esta manera no habría que estar moviendo y cambiando el nombre a los ficheros, además de que pueden crear problemas con las extensiones que tengan.

Bibliografía

[1] R. F. Harrington. Field Computation by Moment Methods. MacMillan, New York, 1968.

[2] R.F. Harrington. Matrix Methods for Field Problems. Proceedings of the IEEE, Vol. 55, No. 2, pp. 136-149, February 1967.

[3] J. H. Richmond. A Wire-Grid Model for Scattering by Conducting Bodies. IEEE Transactions on Antennas and Propagation, Vol. AP-14, No. 6, pp. 782-786, November 1966.

[4] A. W. Glisson, D. R. Wilton. Simple and Efficient Numerical Methods for Problems of Electromagnetic Radiation and Scattering from Surfaces. IEEE Transactions on Antennas and Propagation, Vol. Ap-28, No. 5, pp. 593-603, September 1980

[5] N. C. Albertsen, J. E. Hansen, N. E. Jensen. Computation of Radiation from Wire Antennas on Conducting Bodies. IEEE Transactions on Antennas and Propagation Vol. Ap-22, No. 2, pp. 200-206, March 1974.

[6] L. N. Medgyesi-Mitschanf, D. Wang. Hybrid Methods for Analysis of Complex Scatterers. Proceeding of the IEEE, Vol. 77, No. 5, pp. 770-779, May 1989.

[7] G. Farin. Curves and Surfaces for Computer Aided Geometric Desing: A Practical Guide. Academic Press, 1988.

[8] L. Valle, F. Rivas, M. F. Cátedra. Combining the Moment Method with Geometrical Modelling by NURBS Surfaces and Bézier Patches. IEEE Transactions on Antennas and Propagation Vol. Ap-42, No. 3, pp. 373-381, March 1994.

[9] L. Valle, F. Rivas, M. F. Cátedra. Cálculo del Campo Scattering Producido por Cuerpos de Geometrías Arbitraria Modelados por Superficies NURBS usando el Método de los Momentos. VII Symposium Nacional de la URSI, pp. 910-914, Málaga 23-25 Septiembre de 1992.

[10] F. Rivas, L. Valle, M. F. Cátedra. Análisis de antenas de Hilo mediante el Método de los Momentos sobre Cuerpos de Geometría Modelada por Parches de Bézier. VIII Symposium Nacional de la URSI, pp 704-708, Valencia 22-24 Septiembre de 1993.

[11] L. Valle, F. Rivas, M. F. Cátedra. Electromagnetic Scattering by Arbitrary Shaped Bodies Modelled with NURBS Surfaces Using the Method of Moments. Journees Internationales de Nice sur les Antennes, pp. 35-38, Nice 1992.

[12] https://es.mathworks.com/discovery/matlab-gui.html

Anexo

```
function BotonEjecutar_Callback(hObject, eventdata, handles)
% hObject    handle to BotonEjecutar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
fid = fopen('datos.dat', 'w+')
tipo = get(handles.Tipocal, 'String'); %Cojo las opciones que hay en el menu
a = get(handles.Tipocal, 'Value'); %Número de las opciones
opcion = tipo(a); %Opcion escogida
O = cell2mat(opcion); %Lo convertimos en una sola matriz
fprintf(fid, '%s ! Tipo de calculo a realizar: RCS, RCM, DIA, ACO, DIP o ACP.', O);
Theta = get(handles.Theta, 'String');
T = str2num(Theta);
Phi = get(handles.Phi, 'String');
P = str2num(Phi);
fprintf(fid, '\n%3.1f,%3.1f ! Angulos THETA,PHI.', T, P);
Frec = get(handles.Frec, 'String');
F = str2num(Frec);
fprintf(fid, '\n%.3fE+6 ! Frecuencia.', F);
if (get(handles.Completo,'Value') == 0)
    fprintf(fid, '\n1.0,0.0 ! Metodo hibrido: 0.- completo,1.- visible. Distancia cercano.');
else
    fprintf(fid, '\n0.0,0.0 ! Metodo hibrido: 0.- completo,1.- visible. Distancia cercano.');
end
Iter = get(handles.Iter, 'String');
I = str2num(Iter);
fprintf(fid, '\n%i ! Numero maximo de iteraciones.', I);
Error = get(handles.Error, 'String');
E = str2num(Error);
fprintf(fid, '\n%.1fE-03 ! Maximo error permitido.', E);
Criterio = get(handles.Criterio, 'String');
C = str2num(Criterio);
fprintf(fid, '\n%.1f ! Criterio de cercania entre subdominios.', C);
Curvos = get(handles.Curvos, 'String');
PC = str2num(Curvos);
fprintf(fid, '\n%i ! Divisiones por lambda (parches curvos).', PC);
Planos = get(handles.Planos, 'String');
PP = str2num(Planos);
```

```
fprintf(fid, '\n%i ! Divisiones por lambda (parches planos).', PP);
Radar = get(handles.Radar, 'String');
fprintf(fid, '\n%s ! Tipo de seccion radar o patron de radiacion.', Radar);
Norm = get(handles.Norm, 'String');
N = str2num(Norm);
fprintf(fid, '\n%.f. ! Factor de normalizacion del campo.', N);
CortesT = get(handles.CortesT, 'String');
CT = str2num(CortesT);
fprintf(fid, '\n%i ! Cortes en theta constante.', CT);
CortesP = get(handles.CortesP, 'String');
CP = str2num(CortesP);
fprintf(fid, '\n%i ! Cortes en phi constante.', CP);
IniT = get(handles.IniT, 'String');
IT = str2num(IniT);
IniP = get(handles.IniP, 'String');
IP = str2num(IniP);
Paso = get(handles.Paso, 'String');
Pa = str2num(Paso);
Muestras = get(handles.Muestras, 'String');
M = str2num(Muestras);
fprintf(fid, '\n%.1f,%.1f,%.1f,%i ! Theta, Phi inicial, Paso en Phi, Número de muestras.', IT, IP, Pa, M);
fclose(fid);
%Antes de LeerFichero hacer los pasos previos que tengo apuntados en la hoja
copyfile('datos.dat','datos','f')
 system('Monurbs.exe')
 while isprocess('Monurbs.exe')
 end
LeerFichero
```

```
function Guardar_Callback(hObject, eventdata, handles)
% hObject    handle to Guardar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[archivo, ruta] = uiputfile({'..\TFG Alejandro Gutiérrez pruebas\Guardar\.dat'},'Guardar archivos')
cad1 = strcat(ruta, archivo)
cad2 = strrep(cad1, '.dat', '.geo')
%archivo = uiputfile ('', 'Seleccionar un archivo')
copyfile('datos/datos.dat','Guardar', 'f')
movefile ('Guardar/datos.dat', cad1)
copyfile('datos/geo','Guardar', 'f')
movefile ('Guardar/geo', cad2)
%Llega el fichero con el nombre cambiado pero no tiene ninguna extensión
%Esto lo dejamos como prueba pero no me gusta
```

```
function Pieza_Callback(hObject, eventdata, handles)
% hObject    handle to Pieza (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns Pieza contents as cell array
%        contents{get(hObject,'Value')} returns selected item from Pieza
pieza = get(hObject, 'String'); %Cojo las opciones que hay en el menu
b = get(hObject, 'Value'); %Número de las opciones
eleccion = pieza(b); %Opcion escogida

switch cell2mat(eleccion)

    case 'Placa'
        copyfile('Modelos/Placa','datos','f')
        movefile datos/Placa datos/geo
        figure(2)
        %run('placa')
        system('Monurbs_Modelos.exe')
        while isprocess('Monurbs_Modelo.exe')
        end
        run('modelo')
        hold off

    case 'Esfera'
        copyfile('Modelos/Esfera.dat','datos','f')
        movefile datos/Esfera.dat datos/geo
        figure(2)
        %run('esfera')
        system('Monurbs_Modelos.exe')
        while isprocess('Monurbs_Modelo.exe')
        end
        run('modelo')
        hold off
```

```
        '
    case 'ConoInferior'
        copyfile('Modelos/ConoInferior.dat','datos','f')
        movefile datos/ConoInferior.dat datos/geo
        figure(2)
        %run('ConoInferior')
        system('Monurbs_Modelos.exe')
        while isprocess('Monurbs_Modelo.exe')
        end
        run('modelo')
        hold off

    case 'DosConos'
        copyfile('Modelos/DosConos.dat','datos','f')
        movefile datos/DosConos.dat datos/geo
        figure(2)
        %run('DosConos')
        system('Monurbs_Modelos.exe')
        while isprocess('Monurbs_Modelo.exe')
        end
        run('modelo')
        hold off

    case 'CN235'
        copyfile('Modelos/CN235.dat','datos','f')
        movefile datos/CN235.dat datos/geo
        figure(2)
        %run('CN235')
        system('Monurbs_Modelos.exe')
        while isprocess('Monurbs_Modelo.exe')
        end
        run('modelo')
        hold off


end
```